

spi_tm2fits

User Manual

Version 1.5.1
30 July 2002

Jürgen Knödlseeder
Centre d'Etude Spatiale des Rayonnements
knodlseder@cesr.fr
<http://www.cesr.fr/~jurgen/index.html>

Note to the user

This software has been written to analyse data of the SPI telescope onboard INTEGRAL. Particular care has been taken in making the software user friendly and well documented. If you appreciated this effort, and if this software and User Manual were useful for your scientific work, the author would appreciate a corresponding acknowledgment in your published work.

Contents

1	Introduction	1
2	Getting started	1
3	Parameter file	2
4	Interface definition	5
5	Error codes	8

1 Introduction

The executable **spi_tm2fits** converts INTEGRAL or SPI telemetry in the format of the CESR GSE acquisition system (**PktA11**) into a ISDC compliant data structure of ISDC level **RAW**. Each run of **spi_tm2fits** produces a single science windows, hence to cut a data stream into science window it has to be applied successively to the input data with different input packet limitations (use the task **spi_tmscan** to automatically divide a data stream into science windows).

spi_tm2fits comes with a number of data structure templates that may either replace or supplement existing ISDC templates. Templates that are replaced are ISDC templates that do not reflect the actual status of the data base. Templates that are supplemented concern data that is normally not analysed by the ISDC system, such as CNES GSE packets or sub-assembly memory dumps.

spi_tm2fits has been written in the ANSI C++ language, and requires the libraries **spi_psdlib** and **spi_toolslib**. The actual version is compliant with the ISDC support platform 4.1.

2 Getting started

Before installing **spi_tm2fits**, make sure that the ISDC support platform 4.1 or higher is installed on your system, and that the libraries **spi_psdlib** and **spi_toolslib** are also available (please consult the WWW site <http://www.cesr.fr/~jürgen/isdc> if these libraries are not available on your system).

After downloading the **spi_tm2fits.tar.gz** file, step into a directory that should hold the distribution, move the **spi_tm2fits.tar.gz** file into this directory and type:

```
$ gunzip spi_tm2fits.tar.gz
$ tar xvf spi_tm2fits.tar
```

The first command uncompresses the distribution file, the second unpacks the files.

Before configuration, the distribution needs to be reset to a clean state. To do this, type

```
$ make distclean
```

Then, configure the distribution. It is assumed here that you have previously installed the ISDC support platform, thus you should type

```
$ ~/bin/ac_stuff/configure
```

Finally, build the distribution by typing

```
$ make global_install
```

3 Parameter file

```
#####
#
#           Centre d'Etude Spatiale des Rayonnements
#           (in collaboration with ISDC)
#
#           SPI Pre Processing
#
# -----
#
# File:      spi_tm2fits.par
# Version:   1.5.0
# Component: pp
#
# Author:    Juergen Knoedlseder
#            knodlseder@cesr.fr
#            CESR
#
# Purpose:   Parameter file of the SPI pre-processing executable
#
# History:   1.5.0 18-Jul-2002
#
#####
#
# Path name definition
#=====
tmpath, s, h,          "/x02/tm",,, "TM Base Path"
scwbase, s, h, "/villa-1/jurgen/arc/tst_tm/scw",,, "Science Window Base Path"
cfgbase, s, h,  "/users/jurgen/isdc/templates",,, "Template Base Path"
#
# TM input definition
#=====
nfirst, i, ql,          515,, "First TM file number"
nlast,  i, ql,          515,, "Last TM file number"
pktmin, i, ql, 400174,1,10000000, "First packet to scan in first TM file"
pktmax, i, ql, 447092,1,10000000, "Last packet to scan in last TM file"
header, i, h,          0,, "Header size (in Bytes)"
#
# Science window definition
#=====
revid, s, ql, "9000",,, "Revolution ID"
pointid, s, ql, "0080",,, "Pointing ID (minimum is 0000)"
subid, s, ql, "001",,, "Subdivision ID (minimum is 001)"
scwtype, s, ql, "2",0|1|2,, "Science Window Type (0=pointing, 1=slew, 2=other)"
version, s, ql, "001",,, "Version number"
previd, s, ql, "000000000000",,, "Previous Science Window ID"
#
# Task parameters
#=====
method, s, ql, "PROCESS",SCAN|PROCESS,, "Working method (SCAN/PROCESS)"
bcpid, s, ql, "00000000",,, "Broadcast packet ID"
swbound, s, ql, "UNDEFINED",,, "Reason for Science Window ending"
```

```
#
# ISDC Standard Parameters
#=====
clobber, b, h,    no,,, "Clobber Flag"
mode,      s, h, "ql",,, "Execution mode"
```

The parameters have the following meanings:

- **tmpath** specifies the directory which holds the **PktAll** telemetry files in the CESR GSE format (in this format, telemetry frames are stored as successive blocks of 440 Bytes in a single binary file).
- **scwbase** specifies the Science window base path into which the science window data structures should be written.
- **cfgbase** specifies the absolute directory in which the science window group configuration files will be found. Normally, the directory where the data structure templates are found should be specified.
- **nfirst** specifies the file number of the first **PktAll** telemetry file that should be processed.
- **nlast** specifies the file number of the last **PktAll** telemetry file that should be processed.
- **pktmin** specifies the first packet in the first telemetry binary file (specified by the parameter **nfirst**) that should be processed (the first packet has the packet number 1). A packet is defined as a 440 Bytes block in the telemetry binary file. If **pktmin** = 1, pre-processing starts from the beginning of the telemetry file.
- **pktmax** specifies the last packet in the last telemetry binary file (specified by the parameter **nlast**) that should be processed (packets start from number 1). A packet is defined as a 440 Bytes block in the telemetry binary file. If **pktmax** is larger than the number of available packets, the telemetry file is analysed until the end of the file is reached.
- **header** specifies the number of Bytes in the telemetry binary file header. Earlier versions of the **PktAll** files had a header attached. The actual version do not have a header anymore, hence as default, 0 should be specified.
- **revid** specifies the revolution identifier of the science window that should be generated. This parameter has to be a 4 character string.
- **pointid** specifies the pointing identifier of the science window that should be specified. This parameter has to be a 4 character string. The minimum pointing identifier is 0000.
- **subid** specifies the subdivision identifier of the science window that should be specified. This parameter has to be a 3 character string. The minimum subdivision identifier is 001.
- **scwtype** specifies the science window type. The following types are allowed: 0 is a pointing, 1 is a slew, and 2 is any other science window type.
- **version** specifies the science window version number. This parameter has to be a 3 character string.
- **previd** specifies the previous science window identifier (without version number). This identifier will be stored in the science window grouping table header, and allows for linking of the science windows.
- **method** specifies the telemetry processing method. If **SCAN** is specified, the telemetry file is only scanned and statistics about the contained data types will be dumped into the RIL log file. No **RAW** data will be written. If **PROCESS** is specified, the telemetry file will first be scanned to determine the DAL table size of the resulting **RAW** data structures, and in a second pass, the telemetry will be pre-processed and written into the data structures.

- **bcpid** specifies the Broadcast packet identifier for the telemetry, to be stored in the headers of the **RAW** data structures.
- **swbound** specifies the reason for science window ending. Possible reasons are **SLEW**, **POINTING**, **TIMEOUT**, and **UNDEFINED**.
- **clobber** ISDC standard parameter (see Common User Manual).
- **mode** ISDC standard parameter (see Common User Manual).

4 Interface definition

`spi_tm2fits` creates a science window group of ISDC level **RAW** from the input telemetry files that is compliant with the ISDC data structure definitions. The following HDUs are filled by the executable in the public **eng** branch of the science window structure:

Filename	HDU	Content
<code>eng/raw/intl_stsp.fits</code>	INTL-STSP-SRW	Spacecraft time packet
<code>eng/raw/intl_raw_hk.fits</code>	INTL-PLM.-HRW	Spacecraft PLM packet
	INTL-CDMU-HRW	Spacecraft CDMU packet
	INTL-AOCS-HRW	Spacecraft AOCS packet
	INTL-SVM1-HRW	Service module 1 packets (only SPI parameters)
	INTL-SVM2-HRW	Service module 1 packets (only SPI parameters)
<code>eng/raw/spi_raw_acs.fits</code>	SPI.-FEE0-HRW	ACS group 0 count rates
	SPI.-FEE1-HRW	ACS group 1 count rates
	SPI.-FEE2-HRW	ACS group 2 count rates
	SPI.-FEE3-HRW	ACS group 3 count rates
	SPI.-FEE4-HRW	ACS group 4 count rates
	SPI.-FEE5-HRW	ACS group 5 count rates
	SPI.-FEE6-HRW	ACS group 6 count rates
	SPI.-FEE7-HRW	ACS group 7 count rates
	SPI.-FEE8-HRW	ACS group 8 count rates
	SPI.-FEE9-HRW	ACS group 9 count rates
	SPI.-FEEA-HRW	ACS group 10 count rates
	SPI.-FEEB-HRW	ACS group 11 count rates
	SPI.-OACS-HRW	ACS overall count rates
<code>eng/raw/spi_raw_hk.fits</code>	SPI.-DPE.-HRW	SPI DPE HK (1 second)
	SPI.-001.-HRW	SPI HK (8 seconds)
	SPI.-008.-HRW	SPI HK (64 seconds)
	SPI.-080A-HRW	SPI HK (640 seconds - a)
	SPI.-080B-HRW	SPI HK (640 seconds - b)
	SPI.-480A-HRW	SPI HK (3840 seconds - a)
	SPI.-480B-HRW	SPI HK (3840 seconds - b)
	SPI.-480C-HRW	SPI HK (3840 seconds - c)
	SPI.-480D-HRW	SPI HK (3840 seconds - d)
<code>eng/raw/spi_raw_dhk.fits</code>	SPI.-DHK0-CRW	DIAG housekeeping packet 0
	SPI.-DHK1-CRW	DIAG housekeeping packet 1
	SPI.-DHK2-CRW	DIAG housekeeping packet 2
	SPI.-DHK3-CRW	DIAG housekeeping packet 3
	SPI.-DHK4-CRW	DIAG housekeeping packet 4
	SPI.-DHK5-CRW	DIAG housekeeping packet 5
	SPI.-DHK6-CRW	DIAG housekeeping packet 6
	SPI.-DHK7-CRW	DIAG housekeeping packet 7
	SPI.-DHK8-CRW	DIAG housekeeping packet 8
	SPI.-DHK9-CRW	DIAG housekeeping packet 9
	SPI.-DHKA-CRW	DIAG housekeeping packet 10
	SPI.-DHKB-CRW	DIAG housekeeping packet 11

Filename	HDU	Content
eng/raw/spi_acs_calib.fits	SPI.-ACS.-CRW	ACS calibration data
eng/raw/spi_raw_dump.fits	SPI.-DFMD-HRW	DFEE memory dumps (non ISDC)
	SPI.-ASMD-HRW	ACS memory dump (non ISDC)
	SPI.-PDMD-HRW	PSD memory dump (non ISDC)
eng/raw/spi_raw_cnes.fits	CNES-PDU.-HRW	CNES PDU data (non ISDC)
	CNES-RTU.-HRW	CNES RTU data (non ISDC)
eng/raw/spi_raw_block.fits	SPI.-OBLK-HRW	Photon data block structure (OPER)
	SPI.-EBLK-HRW	Photon data block structure (EMER)
	SPI.-CBLK-HRW	Photon data block structure (CALB)
	SPI.-DBLK-HRW	Photon data block structure (DIAG)
eng/raw/spi_raw_schk.fits	SPI.-SHK1-TMP	Science HK (temporary 1)
	SPI.-SHK2-TMP	Science HK (temporary 2)
	SPI.-SHK3-TMP	Science HK (temporary 3)
	SPI.-SHK4-TMP	Science HK (temporary 4)
	SPI.-SHK5-TMP	Science HK (temporary 5)

The following HDUs are filled by **spi_tm2fits** in the private **spi** branch of the science window structure:

Filename	HDU	Content
spi/raw/spi_raw_oper.fits	SPI.-OSGL-RAW	Single events (OPER)
	SPI.-OPSD-RAW	PSD events (OPER)
	SPI.-OCR-RAW	PSD curves (OPER)
	SPI.-OME2-RAW	Double events (OPER)
	SPI.-OME3-RAW	Tripple events (OPER)
	SPI.-OME4-RAW	Quadruple events (OPER)
	SPI.-OME5-RAW	Quintuple events (OPER)
	SPI.-OMEH-RAW	Multiple events (OPER)
	SPI.-OMP3-RAW	Tripple with PSD events (OPER)
	SPI.-OMP4-RAW	Quadruple with PSD events (OPER)
	SPI.-OMP5-RAW	Quintuple with PSD events (OPER)
	SPI.-OMP6-RAW	Setuple with PSD events (OPER)
	SPI.-OMPH-RAW	Multiple with PSD events (OPER)
	SPI.-OPPS-RAW	Pure PSD events (OPER)
spi/raw/spi_raw_emer.fits	SPI.-EME2-RAW	Double events (EMER)
	SPI.-EME3-RAW	Tripple events (EMER)
	SPI.-EME4-RAW	Quadruple events (EMER)
	SPI.-EME5-RAW	Quintuple events (EMER)
	SPI.-EMEH-RAW	Multiple events (EMER)
	SPI.-EMP3-RAW	Tripple with PSD events (EMER)
	SPI.-EMP4-RAW	Quadruple with PSD events (EMER)
	SPI.-EMP5-RAW	Quintuple with PSD events (EMER)
	SPI.-EMP6-RAW	Setuple with PSD events (EMER)
	SPI.-EMPH-RAW	Multiple with PSD events (EMER)
spi/raw/spi_raw_calb.fits	SPI.-EPPS-RAW	Pure PSD events (EMER)
	SPI.-CCRV-RAW	PSD curves (CALB)
spi/raw/spi_raw_diag.fits	SPI.-DSGL-RAW	Single events (DIAG)
	SPI.-DPSD-RAW	PSD events (DIAG)
	SPI.-DCRV-RAW	PSD curves (DIAG)
	SPI.-DME2-RAW	Double events (DIAG)
	SPI.-DME3-RAW	Tripple events (DIAG)
	SPI.-DME4-RAW	Quadruple events (DIAG)
	SPI.-DME5-RAW	Quintuple events (DIAG)
	SPI.-DMEH-RAW	Multiple events (DIAG)
	SPI.-DMP3-RAW	Tripple with PSD events (DIAG)
	SPI.-DMP4-RAW	Quadruple with PSD events (DIAG)
	SPI.-DMP5-RAW	Quintuple with PSD events (DIAG)
	SPI.-DMP6-RAW	Setuple with PSD events (DIAG)
	SPI.-DMPH-RAW	Multiple with PSD events (DIAG)
	SPI.-DPPS-RAW	Pure PSD events (DIAG)

Additional data structure may be created by **spi_tm2fits**, yet they will not be filled with information.

The following header keywords of the science window group are set by `spi_tm2fits`:

Keyword	Value	Content
-----	-----	-----
INSTRUME	SPI	INTEGRAL Instrument
ERTFIRST		Earth received time start of telemetry
ERTLAST		Earth received time stop of telemetry
SWBOUND		Reason for Science Window ending
PCKSTART		First TM packet
PCKEND		Last TM packet

In addition, the following keywords are added to the `RAW` data structure headers:

Keyword	Value	Content
-----	-----	-----
ERTFIRST		Earth received time start of telemetry
ERTLAST		Earth received time stop of telemetry
PCKSTART		First TM packet
PCKEND		Last TM packet
SSC_BEG		First SSC in data
SSC_END		Last SSC in data
SSC_GAP		Number of SSC gaps in data

All photon data structures will also have the additional keywords:

Keyword	Value	Content
-----	-----	-----
LOBT_SBK		Local OBT start of block
LOBT_EBK		Local OBT end of block
BLCK_NUM		Number of 125 ms patterns in data
BLCK_GAP		Number of 125 ms pattern gaps in data

5 Error codes

The following error codes may be returned by `spi_tm2fits`:

SPI_TM2FITS_ERROR_BASE	-10000 // Error base
SPI_TM2FITS_ERROR_MEM_ALLOC	-10000 // Memory allocation error
SPI_TM2FITS_ERROR_EOF	-10001 // End of file reached
SPI_TM2FITS_ERROR_FILE_ERROR	-10002 // TM file error
SPI_TM2FITS_ERROR_INVALID_BUFFER	-10003 // Invalid buffer